PENGGUNAAN HIBRIDISASI *GENETICS ALGORITHMS* DAN *FUZZY SETS* UNTUK MEMPRODUKSI PAKET SOAL

Rolly Intan
Fakultas Teknologi Industri, Jurusan Teknik Informatika, Universitas Kristen Petra
e-mail: rintan@petra.ac.id

ABSTRAK: Proses penyusunan paket soal (misalnya soal untuk test seleksi masuk universitas) yang diambil dari suatu bank soal, minimal harus memperhatikan dua aspek penting yaitu: tingkat kesulitan dan tingkat diskriminan soal. Semakin tinggi tingkat diskriminan suatu soal, semakin baik soal tersebut dipakai untuk menyeleksi kemampuan peserta test. Permasalahan yang dihadapi adalah bagaimana agar pembuat soal dapat memilih dan menentukan kombinasi soal-soal yang tepat (optimum) sehingga dapat memenuhi tingkat kesulitan dan diskriminan yang dikehendaki. Untuk menyelesaikan masalah ini, diperkenalkan suatu algoritma yang disusun dengan menggunakan hibridisasi metode *Genetics Algorithm* dan *fuzzy sets*. Dari hasil pengujian, didapatkan bahwa penggunaan *fuzzy sets* dan *fuzzy relations* dalam pemilihan kromoson awal akan lebih mempercepat pencapaian *tolerable solutions*. Tetap dibutuhkan *treshould* maksimum jumlah generasi yang dilakukan untuk mencegah *run time* sistem *overflow*. Soal bacaan cenderung memiliki nilai *Fitnest Cost* yang lebih rendah daripada nilai *Fitnest Cost* untuk paket soal bukan soal bacaan.

Kata kunci: *genetics algorithm, fuzzy sets, hibridization method.*

ABSTRACT: *At least, two important factors, discrimination and difficulty, should be considered in determing whether a problem should be in a packet of problems produced for students entrance examination at a university. The higher the discrimination degree of a problem, the better the problem is used to make a selection of participants based on their intellectual capability. How to provide a packet of entrance examination problems satisfying a determined pattern of discrimination and difficulty is a major problem in this paper for which an algorithm, it can be proved that the beneficiary of applying fuzzy sets and fuzzy relation in determing the first chromosome in the process of GA is that the process can reach tolerable solutions faster. Maximum number of generation is still needed as a threshold to overcome the problem of run time system overflow. Generally, the problems in the form of passages tend to have lower fitnest cost.*

Keywords: *genetics algorithm, fuzzy sets, hibridization method.*

PENDAHULUAN

Sebuah lembaga pendidikan (misalnya universitas), setiap tahun pasti mengadakan test masuk untuk melakukan seleksi terhadap calon mahasiswa. Standarisasi tingkat kesulitan paket soal untuk test masuk harus dijaga untuk menghindari terjadinya perbedaan tingkat kesulitan paket-paket soal yang dikeluarkan setiap tahun sehingga mengakibatkan kesulitan dalam menentukan kualitas mahasiswa baru yang diterima dari tahun ke tahun melalui *passing grade*-nya. Pemilihan dan penentuan soal-soal yang akan dikeluarkan dari ribuan bank soal secara manual merupakan masalah yang kompleks.

Oleh karena itu diperlukan adanya sebuah sistem cerdas yang mampu memilih paket soal yang optimal dengan standart (tingkat kesulitan dan diskriminan) yang diinginkan. Untuk merealisasi pembuatan sistem cerdas tersebut, paper ini memperkenalkan suatu algoritma yang merupakan hibridisasi *Genetics Algorithm* (GA) dan *Fuzzy Sets*. Alasan penggunaan GA karena masalah yang dihadapi adalah masalah kombinasi yaitu untuk mencari kombinasi paket soal yang optimal sesuai dengan keinginan *user* (pembuat soal). Disamping itu juga penggunaan *fuzzy logics* dan *fuzzy relations* adalah untuk menyatakan tingkat kesulitan dan diskriminan setiap soal serta menentukan derajat kesamaan (*similarity degree*) antara standart yang diinginkan dengan setiap soal yang akan dipilih. Pemilihan paket soal untuk populasi awal dilakukan melalui seleksi berdasarkan *fuzzy relation* sehingga hasilnya menjadi lebih baik bila dibanding dengan cara pemilihan secara *random*.

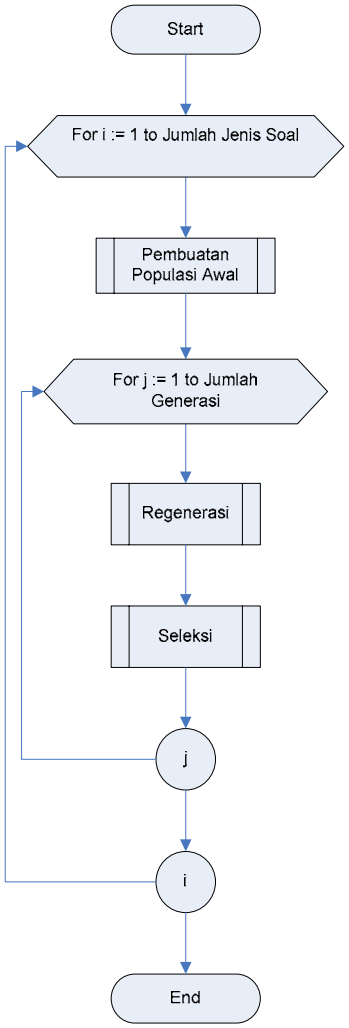
Secara umum, ruang lingkup kerja dari sistem adalah sbb:

a. Tingkat kesulitan dan diskriminan setiap soal ditentukan.
b. Menentukan jumlah jenis soal (mis. Soal Penalaran, Bahasa Indonesia, Bahasa Inggris, dll) yang akan diproduksi.
c. Terdapat soal bacaan untuk jenis soal tertentu, dimana satu kode soal bacaan bisa memiliki lebih dari satu soal.

d. Jumlah soal yang akan dihasilkan, tingkat kesulitan dan diskriminan yang diinginkan di-*input*-kan oleh *user*.
e. Perhitungan *fitness cost* dari suatu kromosom (paket soal) bergantung pada derajat kesamaan antara atribut–atribut (tingkat kesulitan dan diskriminan) yang ada pada soal dengan tingkat kesulitan dan diskriminan yang diinginkan *user*.

ALGORITMA HIBRIDISASI GA DAN *FUZZY SETS*

Algoritma hibridisasi GA dan *fuzzy sets* yang dibuat untuk memproduksi paket soal terdiri dari lima proses utama, yaitu: pengkodean kromosom, pembuatan populasi awal dengan *fuzzy set*, proses regenerasi (mutasi dan *crossover*), proses seleksi dan pengulangan proses regenerasi dan seleksi. Secara garis besar urutan kelima proses tersebut dapat dilihat pada Gambar 1.



Gambar 1. Flowchart Algoritma Hibridisasi GA dan *Fuzzy Sets*

Pengkodean Kromosom

Pengkodean kromosom merupakan proses awal sebelum masuk proses regenerasi. Pengkodean kromosom adalah mengkodefikasi suatu jenis soal tertentu ke dalam model-model kromosom yang didalamnya mengandung beberapa gen. Pengkodean kromosom dilakukan untuk setiap jenis soal, sebagai contoh jenis soal Penalaran, Bahasa Inggris, Bahasa Indonesia, dsb. Kode kromosom ini akan dibedakan berdasarkan inisial atau huruf pertama dari masing-masing kromosom. Sebagai contoh inisial– inisial yang digunakan:
'A': Inisial untuk jenis soal Penalaran,
'B': Inisial untuk jenis soal Bahasa Inggris,
'C': Inisial untuk jenis soal Bahasa Indonesia,

Dimana untuk jenis soal baru berikutnya akan mengikuti urutan dari huruf 'A' – 'Z', jadi bila sudah terdapat tiga jenis soal maka jenis soal keempat akan diberi inisial 'D'.

Pembuatan Populasi Awal Dengan F*uzzy Set*

Pembuatan populasi awal menggunakan konsep *fuzzy set* dan *fuzzy relation* untuk menentukan nilai kesesuaian antara *difficulty* dan diskriminan yang diinputkan user dengan nilai *difficulty* dan diskriminan setiap soal yang ada pada *database* (bank) soal berdasarkan toleransi penyimpangan yang diijinkan (ditentukan).

Yang dimaksud dengan nilai penyimpangan adalah selisih dari nilai yang diinginkan dengan nilai yang terpilih, sebagai contoh apabila tingkat kesulitan yang diinginkan 0.3 sedang tingkat kesulitan soal yang terpilih adalah 0.4 maka nilai penyimpangannya adalah:

$$|0.4\text{-}0.3| = 0.1.$$

Toleransi penyimpangan *difficulty* ($\alpha \in [0,1]$) adalah suatu nilai untuk menentukan batasan maksimum nilai penyimpangan yang diijinkan. Rumus untuk menghitung nilai kesesuaian soal berdasarkan *difficulty* didefinisikan sbb:

$$Diff(\delta(x), \gamma) = \max[0, (1 - \alpha \cdot |\delta(x) - \gamma|)] \qquad (1)$$

di mana:
$\delta(x) \in [0,1]$: Tingkat Kesulitan (*difficulty*) dari soal x.
$\gamma \in [0,1]$: Suatu koefisien *difficulty* yang diinginkan.

$Diff : [0,1] \times [0,1] \rightarrow [0,1]$ adalah suatu fungsi untuk mencari nilai kesesuaian antara *difficulty* dari suatu soal terhadap nilai *difficulty* yang diinginkan. $\alpha \in [0,\infty]$ adalah Toleransi Penyimpangan *Difficulty* yang ditentukan.

Dari persamaan (1), semakin kecil nilai α semakin besar toleransi yang diberikan dan puncaknya jika α=0, maka tidak ada pembatasan toleransi nilai kesesuaian, sehingga berapapun penyimpangan nilai akan diterima (*Diff*=1). Semakin besar nilai α, semakin kecil toleransi yang diberikan dan puncaknya jika α =∞, maka tidak ada toleransi yang diberikan, dimana nilai kesesuaian (*Diff*=1) jika dan hanya jika δ(*x*)=γ.

Hal yang sama dilakukan juga untuk mencari nilai kesesuaian berdasarkan diskriminan yang didefiniskan sbb:

$$Dis(\omega(x),\lambda) = \max[0,(1-\beta\cdot|\omega(x)-\lambda|)] \quad (2)$$

di mana:

$\omega(x) \in$ [-1,1]: Tingkat Diskriminan dari soal *x*.

$\lambda \in$ [-1,1]: Koefisien diskriminan yang diinginkan.

$Dis : [-1,1]\times[-1,1] \rightarrow [0,1]$:suatu fungsi untuk mencari nilai kesesuaian antara diskriminan dari suatu soal terhadap nilai diskriminan yang diinginkan.

$\beta \in [0,\infty]$ adalah Toleransi Penyimpangan Diskriminan yang ditentukan.

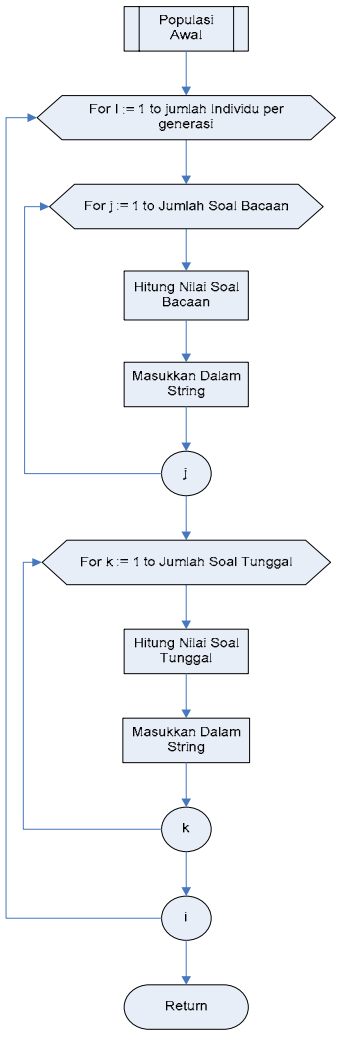
Sebagai contoh: *Difficulty* dari sebuah soal *x*, $\delta(x)$ = 0.2; *Difficulty* yang diinginkan, γ = 0.5; Toleransi penyimpangan, α = 2.5 adalah:

Diff (0.2,0.5) = Max [0, {1- (2.5* |0.2-0.5|)}]
= Max [0, {1-0.75}]
= Max [0, 0.25] = 0.25

Dimana dengan menghitung setiap nilai (*difficulty* dan diskriminan) dari setiap soal yang ada pada *database* maka akan dapat dipilih soal dengan nilai kesesuaian (*Diff* dan *Dis*) terbaik untuk setiap nomor soal pada paket soal yang akan dibuat. Ketentuan–ketentuan untuk membuat generasi awal:

- Karena prioritas tahun keluar merupakan prioritas yang harus (mutlak) dipenuhi, maka seluruh soal pada *database* yang memiliki tahun keluar maksimal yang lebih besar dari inputan *user* akan dibuang / tidak masuk dalam proses perhitungan.
- Setelah terpilih soal–soal yang memenuhi prioritas tahun keluar, maka setiap soal tersebut akan dihitung nilai kesesuaiannya masing–masing.
- Pemilihan pertama dilakukan untuk soal bacaan, dimana soal bacaan dengan nilai tertinggi akan dipilih untuk menjadi individu awal. Sehingga letak soal bacaan selalu didepan pada setiap jenis soal.
- Setelah pemilihan soal bacaan, selanjutnya akan dipilih soal tunggal dengan nilai kesesuaian tertinggi untuk individu awal.
- Untuk individu pertama pemilihan soal dimulai dari urutan nomor soal yang terkecil (*ascendant*) berdasarkan nilai kesesuaian *difficulty* (*Diff* ($\delta(x),\gamma$)) soal.
- Untuk individu kedua pemilihan soal dimulai dari urutan nomor soal yang terkecil (*ascendant*) berdasarkan nilai kesesuaian diskriminan (*Dis*($\omega(x),\lambda$)) soal.
- Untuk individu ketiga pemilihan soal dilakukan dari nomor soal yang terbesar (*descendant*) berdasarkan nilai kesesuaian *difficulty* (*Diff*($\delta(x),\gamma$)) soal.
- Untuk individu keempat pemilihan soal dilakukan dari nomor soal yang terbesar (*descendant*) berdasarkan nilai kesesuaian diskriminan (*Dis*($\omega(x),\lambda$)) soal dari.
- Untuk individu kelima sampai kedelapan diperoleh dari hasil *crossover* individu kesatu sampai individu keempat. Jadi dalam setiap generasi terdapat 8 individu, dimana setiap individu mewakili satu paket soal yang akan dipilih.

Gambar *Flowchart* untuk proses pembuatan populasi awal ini dapat dilihat pada Gambar 2.



Gambar 2. *Flowchart* Pembentukan Populasi Awal

Proses Regenerasi

Proses regenerasi ini adalah proses dimana kromosom-kromosom dalam bentuk *string* yang sudah tersusun dalam proses membuat populasi awal, diolah sedemikian rupa dengan tujuan menghasilkan gen terbaik. Dalam proses regenerasi dilakukan proses perubahan gen-gen generasi sekarang ke gen-gen generasi berikutnya. Hal ini ditandai dengan pembuangan gen-gen generasi sekarang dan hadirnya gen-gen baru yang merupakan anggota generasi berikutnya.

Dalam proses regenerasi, dilakukan dua subproses, yaitu: mutasi, atau *crossover*. Kedua subproses ini tidak dilakukan semuanya pada setiap generasi, namun kedua subproses ini dipilih secara acak untuk dilakukan dengan perbandingan probabilitas dipilihnya subproses mutasi : *crossover* = 1 : 1.

Tahap selanjutnya dilakukan proses seleksi yaitu diadakan pemilihan kromosom terbaik dari setiap generasi. Jika kromoson terbaik dari suatu generasi lebih baik dari kromosom terbaik sekarang, maka terjadi pengisian kromosom terbaik dengan kromosom terbaik pada generasi tersebut (akan dijelaskan lebih detil pada 2.4).

Kromosom terbaik dari proses akan disimpan dan diikutkan terus menerus ke dalam generasi yang berikutnya, hal ini bertujuan untuk menghindari agar kromosom terbaik tidak hilang.

Flowchart bagian proses regenerasi dapat dilihat pada Gambar 3.

Proses Mutasi

Dalam proses ini dilakukan mutasi atau penukaran pasangan gen yang telah dipilih secara random dalam satu kromosom. Penukaran pasangan ini dilakukan pada dua gen dalam suatu kromosom. *Flowchart* proses mutasi dapat dilihat pada Gambar 4.

Proses *Crossover*

Dalam proses ini dilakukan *crossover* atau penukaran bagian gen yang telah dipilih posisinya secara random dalam satu kromosom. Dalam proses ini, perkawinan yang terjadi adalah perkawinan antar seluruh gen dalam suatu generasi.

Syarat–syarat gen yang dapat di *crossover*:

- Kedua Gen harus berada pada nomor awal soal yang sama dan nomor akhir soal yang sama. Hal ini untuk menghindari agar dalam satu kromosom tidak memiliki jumlah soal yang berbeda, karena satu gen dapat mewakili lebih dari satu soal (misalnya: soal bacaan). Sebagai contoh:

Kromosom1

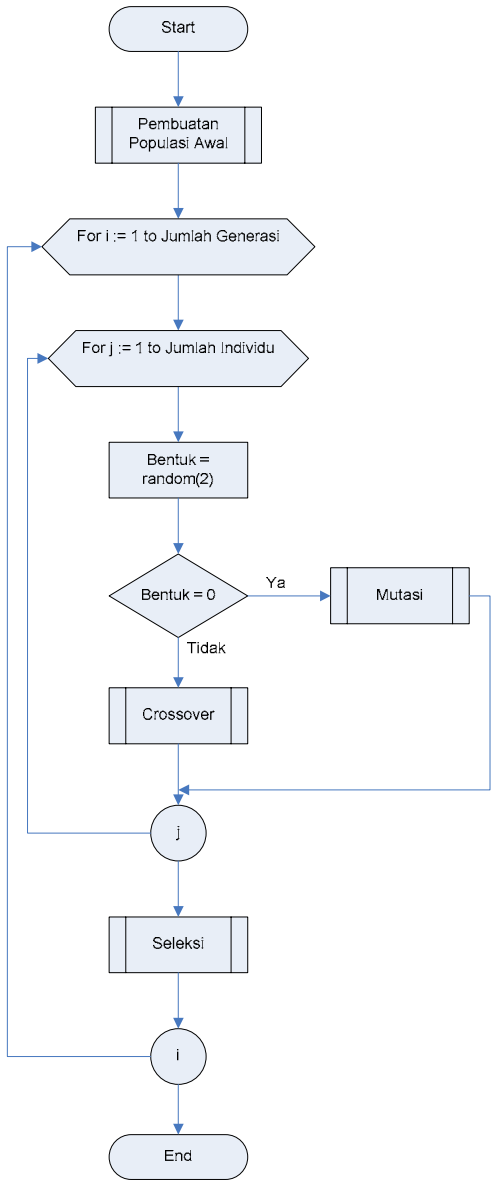
4 Soal	3 Soal	2 Soal	1 Soal

Kromosom 2

4 Soal	5 Soal	1 Soal

Maka diperoleh:
- No Awal Kromosom 1 = {1,5,8,10}
- No Akhir Kromosom 1 = {4,7,9,10}
- No Awal Kromosom 2 = {1,5,10}
- No Akhir Kromosom 2 = {4,9,10}

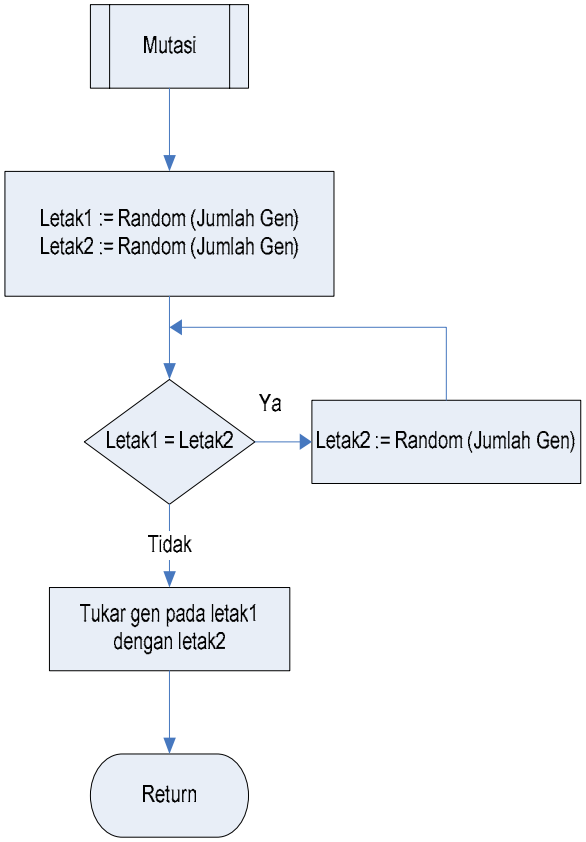
Maka kemungkinan posisi yang ditukar adalah: {1,4} (artinya: No. 1 s/d No. 4), {1,9}, {1,10}, {5,9}, {5,10}, {10}.



Gambar 3. *Flowchart* Proses Regenerasi

- Gen yang akan ditukar tidak boleh kembar dengan susunan gen yang terdapat pada kromosom barunya.
  Sebagai contoh:
  Kromosom 1 : A11,**A15,A13**,A14,A12
  Kromosom 2 : A22,**A23,A21**,A15,A24

  Pada contoh kasus ini tidak boleh dilakukan proses pertukaran karena Gen A15 sudah terdapat pada Kromosom 2.
  Flowchart proses *Crossover* dapat dilihat pada Gambar 6.



Gambar 4. *Flowchart* Proses Mutasi

Proses Seleksi

Proses seleksi adalah proses mencari kromosom terbaik dalam satu generasi. Dimana untuk menentukan suatu kromosom terbaik dapat dilihat dari nilai *fitnest cost*-nya. Rumus untuk menghitung nilai *fitnest cost* didefinisikan sbb:

$$FC(Kr) = C - \sum_{i=1}^{n} (\alpha_i \cdot |\delta_i - \gamma_i|) + (\beta_i \cdot |\omega_i - \lambda_i|) \quad (3)$$

di mana:
FC(*Kr*) adalah nilai *FC* dari suatu kromosom *Kr*.

C: Konstanta = $\sum_{i=1}^{n} (\alpha_i + 2 \cdot \beta_i)$

n: Jumlah Soal

α_i : Koefisien *Difficulty* soal ke-*i*.

β_i : Koefisien Diskriminan soal ke-*i*.

δ_i : Tingkat *Difficulty* soal nomor ke-*i*.

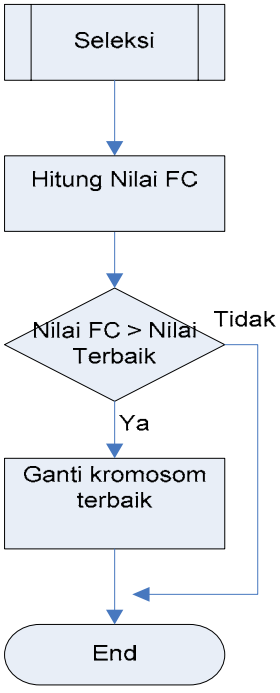
γ_i : Tingkat *Difficulty* yang diinginkan untuk soal ke-*i*.

ω_i = Tingkat Diskriminan soal nomor ke-*i*.

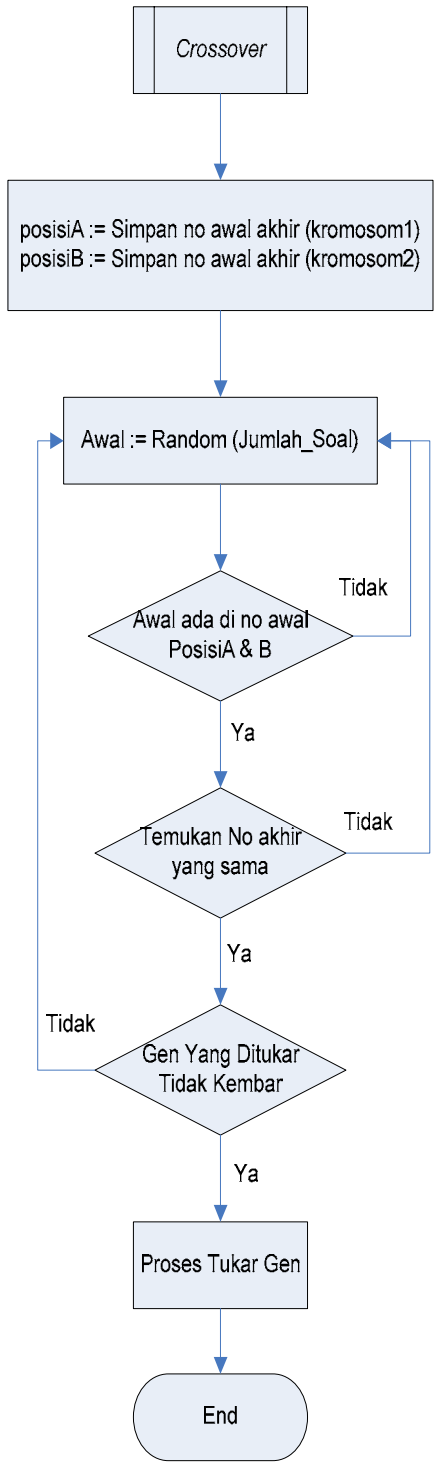
λ_i = Tingkat Diskriminan yang diinginkan soal ke-*i*.

Tujuan dari penggunaan *C* (Konstanta) adalah agar hasil dari penghitungan nilai *fitnest cost* tersebut selalu positif. Hal ini disebabkan karena inisialisasi awal untuk nilai *FC* kromosom terbaik adalah = 0. Apabila ingin membuat paket soal yang lebih ditekankan pada nilai *difficulty* maka *input*an koefisien α yang diperbesar, sebaliknya apabila kita ingin membuat paket soal yang lebih ditekankan pada nilai diskriminan, maka nilai β yang diperbesar. Setelah diketahui nilai FC terbaik maka sesuai dengan metode *elitism*, kromosom terbaik akan disimpan untuk disertakan pada generasi berikutnya.

Flowchart proses Seleksi dapat dilihat pada Gambar 5.



Gambar 5. *Flowchart* Proses Seleksi



Gambar 6. *Flowchart* Proses *Crossover*

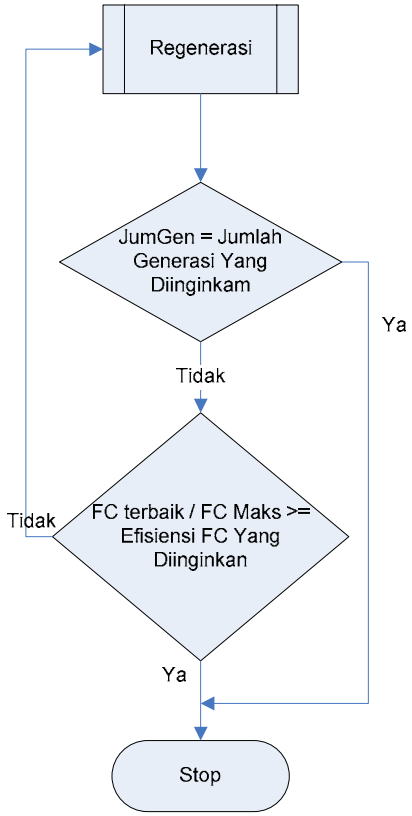
Proses Pengulangan

Dalam aplikasi ini apabila jumlah generasi belum mencapai jumlah generasi yang di*input*kan oleh *user* atau efisiensi dari nilai *Fitnest Cost* belum mencapai keinginan *user* maka akan kembali ke proses regenerasi. Sebaliknya apabila salah satu kondisi terpenuhi maka proses akan berhenti dan kromosom dengan nilai *FC* tertinggi akan dipilih menjadi paket soal. Yang dimaksud dengan efisiensi Nilai *FC* adalah perbandingan antara Nilai *FC* dari kromosom terpilih dengan nilai FC dari maskimal (terbaik). Dimana nilai *FC* maksimal dapat diperoleh dari konstanta *C* pada rumus 3. Sehingga efisiensi nilai *FC* dapat didefinisikan sbb:

$$Eff(Kr) = \frac{FC(Kr)}{C} \times 100\% \qquad (4)$$

dimana *Eff*(*Kr*) adalah efisiensi nilai *FC* dari suatu kromosom *Kr*.

Flowchart proses pengulangan dapat dilihat pada Gambar 7.



Gambar 7. *Flowchart* Proses Pengulangan

KESIMPULAN

Berdasarkan implementasi program yang telah dibuat pada [3], hasil-hasil yang didapat dari pengujian yang dilakukan adalah sebagai berikut:

Keberhasilan dan kecepatan GA untuk mencapai *tolerable solution* (ditentukan oleh *Fitnest Cost* yang diinginkan *user*) sangat dipengaruhi fungsi random, sehingga penggunaan *fuzzy sets* dan *fuzzy relations* dalam pemilihan kromoson awal akan lebih mempercepat pencapaian *tolerable solutions*. Meskipun demikian, tetap diperlukan untuk menentukan proses

regenerasi sampai dengan jumlah generasi tertentu untuk mencegah kemungkinan *tolerable solutions* yang diinginkan terlalu sempurna yang dapat menyebabkan *run time* sistem *overflow*.

Untuk penyusunan paket soal yang terdiri dari soal bacaan cenderung memiliki nilai *Fitnest Cost* yang lebih rendah daripada nilai *Fitnest Cost* untuk paket soal tanpa soal bacaan, hal ini disebabkan karena untuk soal bacaan pengambilan soal dilakukan untuk beberapa nomer sekaligus, dibanding dengan soal tunggal yang hanya diambil untuk satu nomer soal saja.

DAFTAR PUSTAKA

1. G. J. Klir, B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, New Jersey: Prentice Hall, 1995.
2. Goldberg. David E. *Genetic Algorithms in Search, Optimation & Machine Learning*, Addison–Wesley Publishing Companym Inc. 1989.
3. R. Intan, M. Mukaidono, "Approximate Reasoning in Knowledge-based Fuzzy Sets," *Proceeding of NAFIPS-FLINT'02*, New Orleans, June 2002, in press.
4. R. Intan, M. Mukaidono, M. Emoto, "Knowledge- based Representation of Fuzzy Sets," *Proceeding of FUZZ-IEEE'02*, Honolulu, Hawaii, May 2002, pp. 590-595.
5. T. Limantono, *Perencanaan dan pembuatan program untuk memproduksi paket soal test masuk mahasiswa baru dengan menggunakan Metode Genethic Algorithm dan fuzzy set*, Skripsi, Jur. Informatika, UK. Petra, 2005.